IFN680 Assignment Two

(Machine Learning)

Siqi Zhang n9077049

Tinghan Chen n9335757

Overview

There are over 50 thousands lines and 54 columns (12 features) in the dataset Target is forest type, which will be classified into 7 categories. This report describes data preprocessing procedure and analyses the classification performance of Naïve Bayes, Decision Tree, Nearest Neighbors and SVM.

Preprocessing

The dataset is loaded from the file forest_data.pickle. No missing value is detected. Data features and target values are stored into two numpy arrays. In order to ensure the randomness， we shuffled the data. As there are large ranges of variation of data features, we normalized data between 0 and 1 after shuffling. Only first 10 columns need to be normalized because the other values are binary.

We classify the data into two categories: train_data & train_target (top 80% of the data features and target values) and test_data & test_target (rest 20%). We use cross validation (10-fold) to evaluating estimator performance and the function generates a certain proportion of validation_data randomly from train_data. Finally, the shuffled sequence is loaded in to a txt file. Train data and test data will be generated according to the sequence to keep data same for each model.

Naïve Bayes

There are three main types of Naïve Bayes: Gaussian Naïve Bayes, Multinomial Naïve Bayes and Bernoulli Naïve Bayes. Gaussian Naïve Bayes is suitable for continuous values and Bernoulli Naïve Bayes is suitable for binary values. There are 44 binary columns in the dataset. However, these columns contain only 2 features out of 12. Hence, we test the data in both Gaussian Naïve Bayes and Bernoulli Naïve Bayes.

We used train_data and train_target to build the Naïve Bayes model and test_data is to predict the classifier. The classification reports for Gaussian Naïve Bayes and Bernoulli Naïve Bayes are shown in table 1 and table 2.

	precision	recall	f1-score	support
1	0.15	0.02	0.04	42299
2	0.83	0.01	0.02	56697
3	0.29	0.39	0.33	7216
4	0.07	1.00	0.14	543
5	0.02	0.71	0.04	1889
6	0.13	0.06	0.08	3431
7	0.14	0.86	0.24	4128
avg / total	0.48	0.09	0.06	116203

Table 1 Classification report of Gaussian Naïve Bayes classifier

	precision	recall	f1-score	support
1	0.64	0.48	0.55	42299
2	0.65	0.76	0.70	56697
3	0.60	0.87	0.71	7216
4	0.58	0.40	0.47	543
5	0.25	0.07	0.11	1889
6	0.23	0.23	0.23	3431
7	0.64	0.56	0.60	4128
avg / total	0.63	0.63	0.62	116203

Table 2 Classification report of Bernoulli Naïve Bayes classifier

According to table 1 and table 2, Bernoulli Naïve Bayes classifier has higher precision and recall rate than that of Gaussian Naïve Bayes classifier, and the gap is obvious in recall. Hence, we build Bernoulli Naïve Bayes classifier.

[[20289	20642	32	0	49	181	1106]
[9681	43226	1592	13	192	1815	178]
[0	299	6288	101	44	484	0]
[0	0	285	217	0	41	0]
[191	1331	174	0	134	59	0]
[74	378	2037	41	117	784	0]
[1287	496	13	0	0	0	2332]]

Table 3 Confusion matrix of Bernoulli Naïve Bayes classifier

Values in Confusion Matrix (i,j) refer to the observations should be in category i but predicted to be in category j. According to table 2 & table 3, there is a great variability in recall rate of each category. Category 5 is least likely to be recalled (7% recall rate). Category 5 and Category 6 have the worst precision that only over 20% data predicted correctly.

With Naïve Bayes, Category 2 is least likely to be classified because category 2 data falls in all other categories. In comparison, Category 4 is the easiest to be identified. Category 3 has the lowest purify that contains noisy data from all the other categories.

Decision Tree

First we need to find the best parameter for max_depth. We calculate precision score with 10-fold cross validation and accuracy scoring. The score values of depth 30 to depth 60 are generated by matplotlib as shown in figure 1 & figure 2. It can be seen that the Decision Tree with depth 42 has the highest score value.

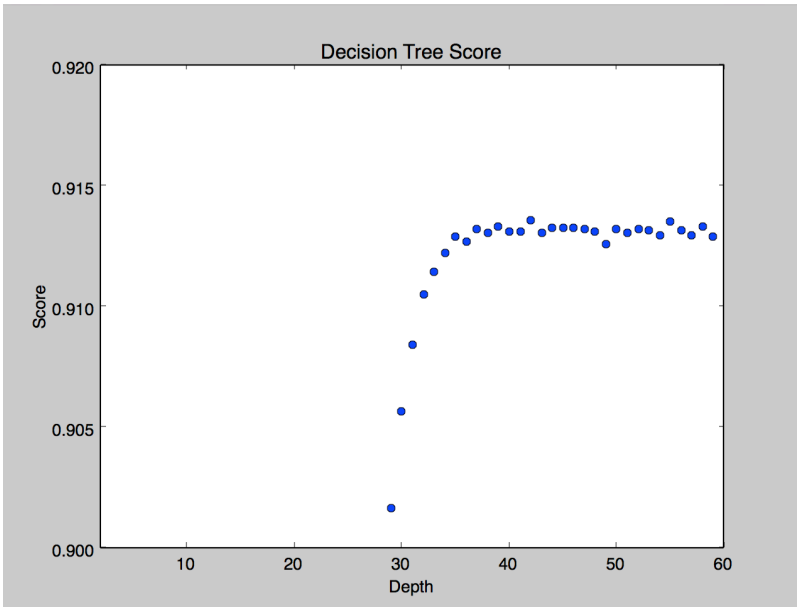

Figure 1 Decision Tree precision scores of depth 30 to depth 60

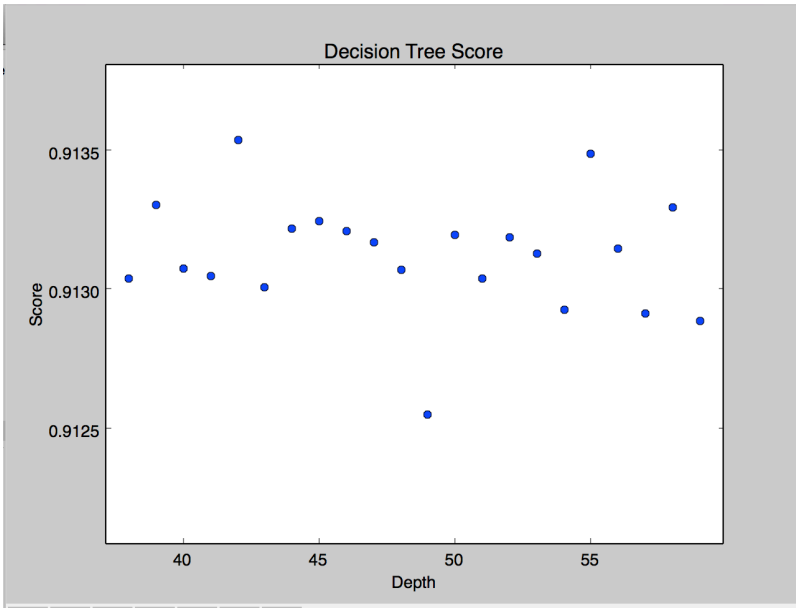

Figure 2 Zoomed decision Tree precision scores

We evaluated the performance with test_data and test_target. The classification report and confusion matrix of the classifier performance are shown in table 4

	precision	recall	f1-score	support
1	0.91	0.91	0.91	42299
2	0.93	0.93	0.93	56697
3	0.91	0.91	0.91	7216
4	0.82	0.82	0.82	543
5	0.78	0.80	0.79	1889
6	0.83	0.83	0.83	3431
7	0.94	0.94	0.94	4128
avg / total	0.92	0.92	0.92	116203

```
[[38663  3324     6     0    68    17   221]
 [ 3427 52469   215     2   326   211    47]
 [    5   230  6579    67    22   313     0]
 [    0     3    64   447     0    29     0]
 [   54   295    21     0  1507    12     0]
 [   28   205   326    29    12  2831     0]
 [  210    56     0     0     0     0  3862]]
```

*Table 4 Classification report of Decision Tree classifie*r

Decision Tree classifier has a good average precision and recall rate of both 92%. There are no significant low values in precision and recall of each category. Most data is predicted in the correct categories. Both precision and recall of each category are similar. However, except Category 7, each category contains many groups of noisy data from other categories. The purity of the classification of this model is not high.

Nearest Neighbors

We build a GridSearchCV object to find the best parameters (n_neighbors and p) for Nearest Neighbors classifier. There are several data features related to distance, so we need to consider p values to find the best method to calculate distance. We set n_neighbors ranges from 2 to 6 and p ranges from 1 to 3. The result shows that n_neighbors=3 and p=1 perform best which means the Nearest Neighbor classifier has 3 neighbors and using manhattan_distance to calculate distance between dots.

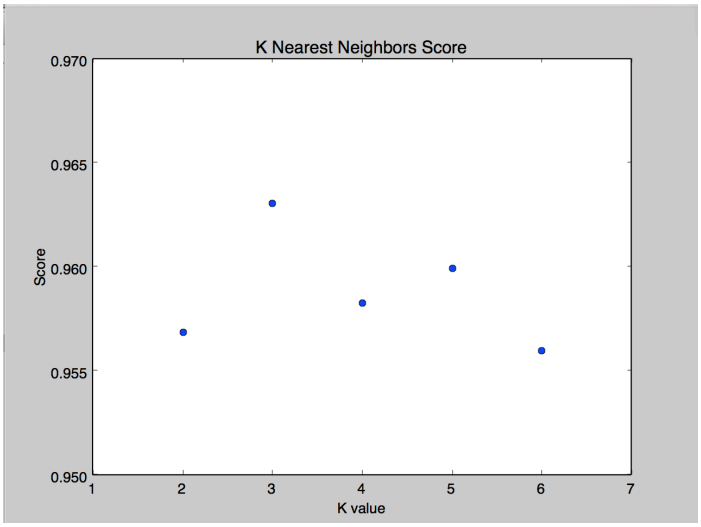

Figure 3 Nearest Neighbors precision scores of different k values

Figure 3 shows the distribution of accuracy scores of different k(n_neighbors) value. There is a trend of falling in score after k value of 3. Thus, k value of 3 will have the peak value with k increasing.

	precision	recall	f1-score	support
1	0.97	0.97	0.97	42299
2	0.97	0.97	0.97	56697
3	0.95	0.96	0.95	7216
4	0.90	0.82	0.86	543
5	0.90	0.89	0.90	1889
6	0.93	0.92	0.93	3431
7	0.97	0.97	0.97	4128
avg / total	0.96	0.96	0.96	116203

```
[[40824  1315     1     0    31     9   119]
 [ 1267 55099   100     0   140    67    24]
 [    4   126  6909    35    13   129     0]
 [    0     4    61   447     0    31     0]
 [   33   145    16     0  1689     6     0]
 [    8    75   167    14     3  3164     0]
 [  110    17     0     0     3     0  3998]]
```

Table 5 Classification report & Confusion matrix of kNearest Neighbors classifier

According to table 5, kNearest Neighbors classifier has a perfect precision and recall rate, which is as high as 96%. There are no significant low values in precision and recall of each category. The overall data purify is slightly higher than that of Decision Tree. Category 4 and Category 7 are relative purer categories.

Support Vector Machines

We build a GridSearchCV object to find the best parameters (kernel and C) for

SVM classifier. Kernel could be linear or rbf and C ranges from 10^-4 to 10^4. As the computation time is long, we only extract the first 10000 lines from train data to build classifier. The result shows that parameters with kernel=linear and C=10^4 performs best. However, figure 4 shows that there is an increasing trend for all C values, which indicates the dataset has a large margin and the higher C is, the better performance will be. We could not guarantee the score in C=10^4 will be still be the peak value when C is larger than 10^4.

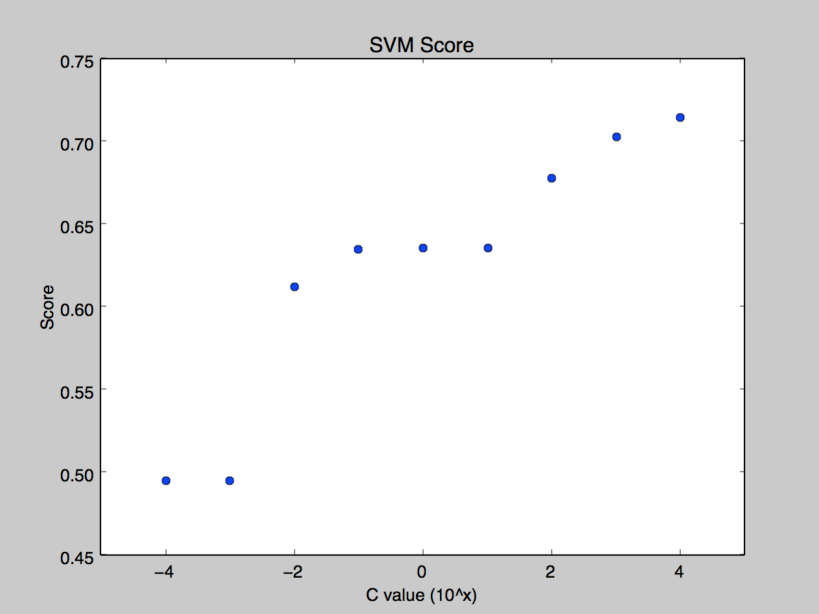

Figure 4 SVM accuracy scores of different c values

	precision	recall	f1-score	support
1	0.71	0.67	0.69	42299
2	0.73	0.81	0.77	56697
3	0.68	0.77	0.72	7216
4	0.61	0.41	0.49	543
5	0.42	0.06	0.10	1889
6	0.45	0.29	0.35	3431
7	0.71	0.53	0.60	4128
avg / total	0.71	0.72	0.71	116203

```
[[28170 13254     8     0    24    18   825]
 [ 9446 45975   756     0   107   330    83]
 [    0   758  5560    95     9   794     0]
 [    0     4   272   224     0    43     0]
 [   26  1679    73     0   109     2     0]
 [    2   911  1483    47     8   980     0]
 [ 1919    24    13     0     0     0  2172]]
```

Table 6 Classification report & Confusion matrix of SVM classifier

Table 6 shows the Classification report and Confusion matrix of SVM classifier of C=10^4. The precision and recall rate are over 70% but recall rate of Category 5

is quite bad. Purify of the categories is quite well. As the classifier is generated from only 10000 lines of data, the table cannot describe actual performance of SVM.

Summary

Nearest Neighbors has the best accuracy performance among the four models, and follows by Decision Tree. Naïve Bayes performs worst. Due to the limited size of train data, the performance of SVM will be better than the evaluation.

Completeness

Implemented and worked

- Loaded dataset, shuffled the data and stored the random sequence to a file
- Normalized data
- Classified data into train data and test data
- Built Naïve Bayes classifier, Decision Tree classifier, kNearest Neighbor classifier and SVM classifier
- Determined best parameters for Decision Tree classifier and kNearest Neighbor classifier
- Evaluated the first three classifiers from Classification report and Confusion matrix

Implemented but does not worked

- Determined best parameters for SVM classifier
- Evaluated the actual performance of SVM classifier